Knowledge Graph-based Question Answering with Large Language Models

Moein Shirdel[1]

Abstract

The integration of Large Language Models (LLMs) with Knowledge Graphs (KGs) offers a novel approach to enhancing question-answering (QA) systems. This integration may address key challenges with LLMs, such as outdated information and the hallucination of responses. This project explores the usefulness of a Retrieval-Augmented Generation (RAG) technique, where an LLM uses structured knowledge retrieved from a knowledge graph to respond to user queries. By taking advantage of the linguistic proficiency of LLMs and populating its context window with the most relevant information, this system aims to analyze a possible enhancement in the response quality of question-answering systems. The developed system primarily focuses on queries related to movies from the MetaQA dataset. The results demonstrate great performance in single-hop query scenarios and highlight areas for improvement in multi-hop settings, where the model encounters challenges in providing coherent responses from KG-retrieved knowledge pieces. The complete code and system implementation for this project is available on GitHub[1].

1. Introduction

1.1. Problem overview (Q1)

In recent years, the intersection of artificial intelligence (AI) and natural language processing (NLP) has witnessed remarkable advancements, particularly with the introduction of large language models (LLMs) and their integration into various applications. Among these applications, question-answering (QA) systems have caught attention due to their potential to facilitate information retrieval and comprehension in diverse domains. In this context, the use of knowledge graphs (KGs) as a structured representation of knowledge, combined with LLMs, can appear as a promising approach to enhancing QA systems' performance.

There was an ongoing debate on the usability and effectiveness of knowledge graphs in the era of LLMs, and many research papers have analyzed the possibility of replacing KGs, considering the ongoing trend and vast abilities of LLMs(Sun et al., 2023). However, many companies and research teams have recently discovered the power of KGs, especially when used alongside LLMs. Such efforts, alongside keynotes made by industry leaders and research papers published recently, show a potential shift towards unifying LLMs knowledge graphs(Pan et al., 2024).

1.2. Problem importance (Q2)

Since LLMs are emerging as primary tools in many tasks, it is of significant importance to maximize their usability and enhance their performance. LLMs' usage presents two main challenges: First, they might respond based on out-of-date information, and the computational cost and complexity of re-training LLMs on brand-new knowledge is significantly high. Secondly, their answers may contain false or non-factual information (an issue known as LLM hallucination), which is inevitable given the instinct of LLMs as generative models. Given the high cost and complexity of re-training LLMs and the demand for current and accurate responses, the significance of implementing an open-book question-answering (QA) approach becomes evident.

This is where the Retrieval-Augmented Generation (RAG) technique comes in, a process in which the LLM is provided with an external dataset and asked to retrieve information in response to a user query only based on the given dataset. This approach has been proven to reduce LLM hallucinations and also to use the linguistic power of LLMs to retrieve up-to-date information from the external dataset. Through this approach, users of LLMs can enhance the reliability and relevance of generated responses, and maximizing the utility of these powerful language models consequently.

1.3. Gaps in previous methods (Q3)

In recent years, there has been significant breakthroughs in the development of chatbots and AI-powered tools focused on question-answering tasks. Many of these approaches leverage graphs and their associated knowledge bases to

[1]University of Waterloo, Waterloo, ON, Canada. Correspondence to: Moein Shirdel <mshirdel@uwaterloo.ca>.

[1]https://github.com/moeinsh78/KGQA-cs886

fulfill their objectives. While numerous research papers have proposed promising models for Knowledge Graph-based Question-Answering (KGQA), they often fall short in comparison to LLMs in terms of linguistic proficiency. Consequently, the current emphasis in the field of QA tasks lies in maximizing the potential of LLMs and fine-tuning them to excel in such tasks.

Among the techniques employed in the area of LLMs, the RAG technique is a well-known method utilized in many different applications. RAG entails presenting the LLM with the entire dataset along with the query or the primary retrieval target from the dataset. Nonetheless, the context windows of language models are limited, and in some applications and use cases, fitting all the relevant information in the context window will be impossible due to the volume of data. Consequently, when disparate pieces of information scattered across multiple documents are required, the LLM may struggle to provide a coherent response due to the inability to encompass all necessary knowledge within a single context window.

Additionally, the baseline RAG fails in tasks that require an understanding of the overview of the semantic concepts discussed in a large data collection. In such scenarios, alternative data representation structures, such as graphs, have emerged as viable solutions. These structures, when combined with LLMs, offer a robust framework for responding to user queries, as explored in this project.

More details on existing work in this context is provided in section 2.

1.4. Our solution (Q4)

The idea of integrating LLMs with Knowledge Graphs to **provide the LLM with the most relevant context window** has recently drawn extensive attention. Microsoft has recently proposed GraphRAG, a tool that uses LLMs to generate a KG based on the given private dataset (used for RAG), and utilizes graph ML approaches to respond to a user query. However, since they have not published a paper on this work, this problem still appears to be open.

In this work, the idea is to refer to a knowledge graph to retrieve a pool of information that are the most likely to contribute to the final answer generated by the LLM in response to a user query. When the LLM is provided with a set of textual documents (as in baseline RAG), the details refering to same topics or same entities might be distributed anywhere in those documents. On the other hand, a knowledge graph organizes all relevant information about a particular subject or idea into one central node or nearby nodes. Therefore, our system aims to leverage the organized structure of the knowledge graph. It focuses on extracting pertinent information from the knowledge graph after a user query and then guiding the LLM to reason based on the question and KG-retrieved information.

1.5. Arising research questions (Q5)

Aside from the underlying structure behind the decision-making process of large language models, the explainability of these models is a crucial requirement. Explaining the reasoning behind the derivation of answers in each model and increasing their transparency helps experts to trust models and understand their decisions more easily. One of the main reasons that the author chose this topic was to get a sense of the plausibility of integrating LLMs and KGs in a QA task, and then applying some explainability functionalities to the knowledge graph in order to identify the most crucial nodes, edges, and graph regions in the decision-making process. Other aroused researched questions need more elaboration on the approach taken for this project. Hence, more details on this section are included in section 7.

2. Related Work

In this section, our purpose is to point out some of the most similar approaches taken in the context of question-asnwering systems. Firstly, (Zhang et al., 2018) presents a novel deep learning architecture and an end-to-end variational learning algorithm, which aims to overcome challenges in question answering systems leveraging knowledge graphs. The paper addresses issues such as noisy question expressions and the need for multi-hop reasoning. It effectively matching noisy question entities to the knowledge graph and conducting complex reasoning tasks. This work has introduced the MetaQA dataset and knowledge graph, which has been broadly used in our project.

(Saxena et al., 2022) discusses the utilization of encoder-decoder Transformer models as a scalable solution for knowledge graph embedding (KGE). KGEs are introduced as low-dimensional vectors that aim to represent the entities in a KG in the vector space. Traditionally, KGE models where large models, especially when applied to real-world graphs with lots of entities. This work shows the importance of the integration of stage pipelines for downstream tasks, and achieves promising results for KG link prediction and question answering over incomplete KGs, which is amongst the most significant tasks in the realm of Knowledge Graphs.

On the other hand, in the context of multi-hop decision-making with LLMs, (Feng et al., 2023) explains the challenges faced by LLMs like ChatGPT and GPT-4, particularly their limitations in domain-specific knowledge and the occurrence of hallucinations during inference. The paper proposes a paradigm called Knowledge Solver, which uses the inherent generalizability of LLMs to search for essential knowledge from external sources. This approach involves designing a simple yet effective prompt to guide LLMs through a multi-hop decision sequence, enabling them to search for knowledge in a zero-shot setting. Importantly, KSL enhances the explainability of LLMs by providing complete retrieval paths, and interprets their reasoning processes.

Additionally, (Abu-Rasheed et al., 2024) discusses the integration of knowledge graphs (KG) as a resource to enhance the precision and relevance of explanations generated by large language models (LLMs) in personalized education settings. It shows the necessity of providing comprehensible explanations alongside learning recommendations to improve learner understanding and engagement. By leveraging semantic relations within the KG, the proposed approach aims to mitigate the risk of model hallucinations and inaccuracies while maintaining the intended learning context. The idea of using knowledge graphs to provide the LLM with a relevant contextual information is also used in this project.

Finally, (Wu et al., 2023) discusses the implementation and evaluation of a KG-to-Text augmented language model approach for knowledge graph question-answering (KGQA). Despite the competitive performance of large language models (LLMs) on knowledge-intensive tasks, they often struggle with long tail knowledge. The proposed approach aims to bridge this gap by transforming KG knowledge into textual statements and enhancing the LLM's ability to comprehend the knowledge effectively. Unlike previous approaches, which focus solely on retrieving KG knowledge to enhance LLM prompting, this method prioritizes the verbalization of KG knowledge to better align with textual representations. Their novel approach was used in setting up descriptions for our knowledge graph edges in this project.

3. System Architecture

The overall idea of building such a system is to utilize large language models' proficiency in linguistics to answer real-world questions. Initially, the author picked a knowledge graph representing information about movies, their actors, directors, writers, and details such as their release year. Further elaboration on this knowledge graph can be found in section 4.1.

The aim was to establish a system capable of responding to user queries related to movies based on the provided knowledge graph. In this context, the system identifies the segment of the knowledge graph that finds it the most likely to contain the requested information based on the user's query. Subsequently, all relevant information within that segment is retrieved and put together in the shape of a document.

However, the challenge lies in organizing the retrieved data and identifying the most relevant chunks of information in that neighborhood to formulate an accurate response to the question. Large Language Models emerged as an appropriate choice for this task. The author picked gpt-3.5-turbo to utilize as the LLM for this project due to its proficiency in natural language understanding. A system prompt was designed to instruct the LLM on the output formatting and to describe the task. In specific, the LLM was asked to interpret the question (query) and locate its answer(s) within the provided information document. It is important to prohibit the LLM from using its own knowledge and instruct it to solely reason based on the information retrieved from the knowledge graph.

The LLM is asked to form its response (which may contain multiple answers) as a list for subsequent comparison with the ground truth. Figure 1 shows an overview of the modules forming the system's architecture.

4. Implementation

4.1. Dataset

For this project, the MetaQA dataset (Zhang et al., 2018) was chosen by the author. This dataset consists of a movie ontology derived from the WikiMovies Dataset, represented in the shape of a knowledge graph, and three sets of question-answer pairs written in natural language: 1-hop, 2-hop, and 3-hop queries.

The main and most important characteristic of such a collection is its alignment between questions and the knowledge graph, enabling answers to be retrieved directly from the graph. Additionally, questions are categorized into three groups based on the number of hops required to traverse the graph nodes to locate the correct answer. Consequently, unnecessary expansion of node neighborhoods is avoided, and the exploration depth is set according to the number of hops needed.

In this dataset, a file under the name "kb_entity.txt" displays node labels in the knowledge graph, helping in finding the relevant region of the graph for answer retrieval. Moreover, a file named "kb.txt" describes the edges of this knowledge graph, with each line representing a triple: (head, relation, tail), showing a connection between nodes "head" and "tail" under the labeled edge "relation".

While the dataset includes both training and test set questions, the author utilized the test set for evaluating the performance of the LLM, as this project leverages pre-trained language models to answer questions. The questions along with their expected answers (ground truth) are accessible in the "x-hop/vanilla/qa_test.txt" directory, along with the paths required to find their answers (e.g., Movie to Director). However, these patterns were not used, and the LLM was tasked with leveraging its linguistic abilities to determine

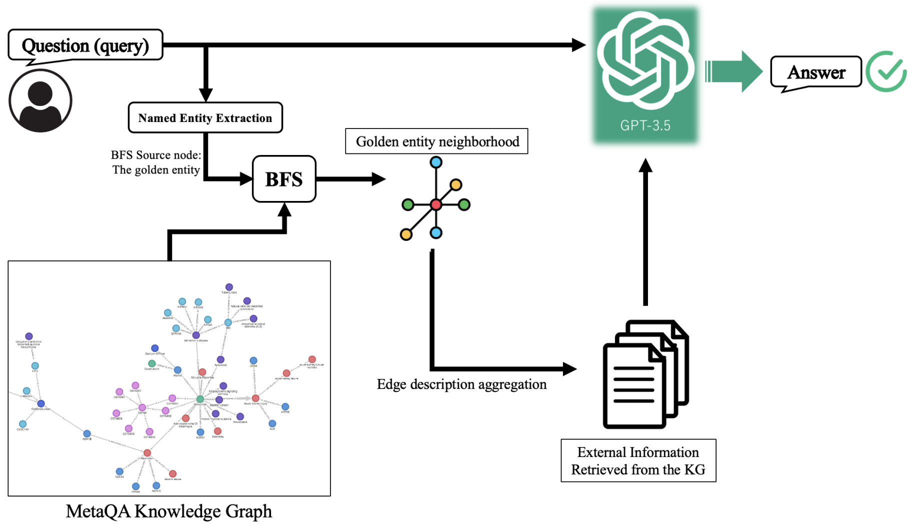

Question (query)
Named Entity Extraction
BFS Source node: The golden entity
BFS
Golden entity neighborhood
GPT-3.5
Answer
Edge description aggregation
External Information Retrieved from the KG
MetaQA Knowledge Graph

Figure 1. An overview of the modules used to construct this system, and the flow from receiving a user query until getting gpt-3.5-turbo's response based on the information retrieved from the underlying knowledge graph.

the correct path autonomously.

4.2. Knowledge Graph Implementation

The author utilized NetworkX (Hagberg et al., 2008) framework to construct a **multi-graph** and represent the knowledge graph. The rationale behind choosing a multi-graph was that a pair of entities might hold multiple connections, each denoting distinct relation types. For instance, in cases where a person appears as both the director and actor in a movie, it was essential to account for each unique edge between the same nodes separately.

As described in the "kb_entity.txt" file, the knowledge graph contains 43,233 entities representing movies, actors, release years, genres, and so on. These entities are connected through 134,741 edges of different types. One end of each edge invariably corresponds to a movie. In other words, all edges point out a characteristic of a movie (e.g., its actors and directors). All edges had certain types, being divided into different categories based on the characteristics of the movie that they were showing. However, it may be difficult for the LLM to reason solely based on the label of each edge and make sense of it. For example, the triple (Inception, starred_actors, Tom Hardy) is just a notation of this edge. In order to provide meaningful information for the LLM, the author assigned a description to each edge, representing the meaning of the edge in natural language, which is specific to the edge type. In the above-mentioned example, the edge description would be as follows: "Actor Tom Hardy starred in Movie Inception."

The knowledge graph contains nine different edge types, the distribution of which is depicted in Table 1 alongside their description format.

Table 1. Distribution of different edge types in MetaQA's knowledge graph and their designated description

Edge Label	Count	Description
X, directed_by, Y	15,966	Movie X is directed by y
X, starred_actors, Y	33,735	Actor Y starred in Movie X
X, written_by, Y	19,543	Movie X is written by Y
X, has_genre, Y	15,895	Movie X has genre Y
X, in_language, Y	3,482	Movie X is in Y language
X, release_year, Y	16,726	Movie X was released in Y
X, has_tags, Y	28,944	Movie X has the tag Y
X, has_imdb_rating, Y	328	Movie X is rated Y in imdb
X, has_imdb_votes, Y	122	Movie X is voted Y in imdb
Total	134,741	-

4.3. Knowledge Graph Information Retrieval

The processes outlined in this stage occur during query time, where the knowledge graph is fully constructed and ready to respond to user inquiries. Upon receiving a query, ideally, we would like to identify the regions of the knowledge graph that hold the necessary pieces of information to answer the received query, aggregate them, and put them into the LLM alongside the question and ask the LLM to answer the question. However, finding this region might be challenging as it requires scanning the whole knowledge graph once and ranking the most relevant nodes to explore.

Multiple approaches can be taken to locate such nodes or regions. One of them is to ask the LLM itself to identify the proper region, which needs to provide information about the knowledge graph and every node to the LLM to find the candidate node to explore, and it seems demanding for this task considering the number of entities. Alternatively, examining the dataset reveals that each question typically references exactly one named entity, with the surrounding region in the knowledge graph containing the requisite information. Such a dataset enables the use of methods such as Named Entity Recognition or simply extracting the named entity from the given query. The latter approach was adopted in this project as developing a scalable method to identify the optimal KG region was beyond its scope.

Another approach involved computing the similarity score between the query and all knowledge graph entities to explore the most similar one. In this regard, we need vectors representing graph entities and user queries. Hence, the author utilized DistilBert pre-trained embeddings (Sanh et al., 2019) to encode the query and entity labels (since the labels were the only available description for each entity). Using cosine similarity, similarity scores were calculated, and the most similar entity labels (top k where k=1,3,10) were retrieved. However, in most cases, the retrieved nodes were irrelevant to the query, and the embeddings did not work properly due to a lack of information about entities in their labels. Such a method might hold reasonable prospects as a scalable method in datasets where proper descriptions of entities are available. Using short labels may not end up properly capturing the relevancy.

Following the identification of the optimal entity for exploration, the graph is traversed using the Breadth-First Search (BFS) method to collect descriptions of the edges surrounding it. The BFS algorithm is performed to a depth equal to the number of hops needed. Each node expansion entails appending the descriptions of its connecting edges to the edge description list. Traversing the KG from the golden entity to the desired depth yields the edge description list, serving as the knowledge base from which we anticipate the LLM to retrieve the query's answer.

The graph shown in Figure 2 results from traversing the KG starting from the node representing Actor "Jean Rochefort", which provided the LLM with the necessary information to answer a question. The reasoning path taken by the LLM is demonstrated on the graph.

4.4. LLM Prompting

One of the main challenges of prompting large language models is providing accurate and insightful system message. This message should explain the task, describe how the input will be given, and how the output should look like. The main focus on the designated message was to instruct the LLM to retrieve the answer only from the provided information. The system message is the same for every query, and can be found in the "promptLLM.py" file.

The prompt would be specific to each query. The prompt consists of two parts: first, the edge description gathered from the knowledge graph, and second, the question itself. The prompting did not follow any specific guideline, and was optimized through a try-and-fail approach. Since we are dealing with an LLM, inconsistency is always an issue. Although the response format was specified in the system prompt, there were limited instances of formatting issues which were further fixed in order to avoid any inconvenience in evaluating system performance.

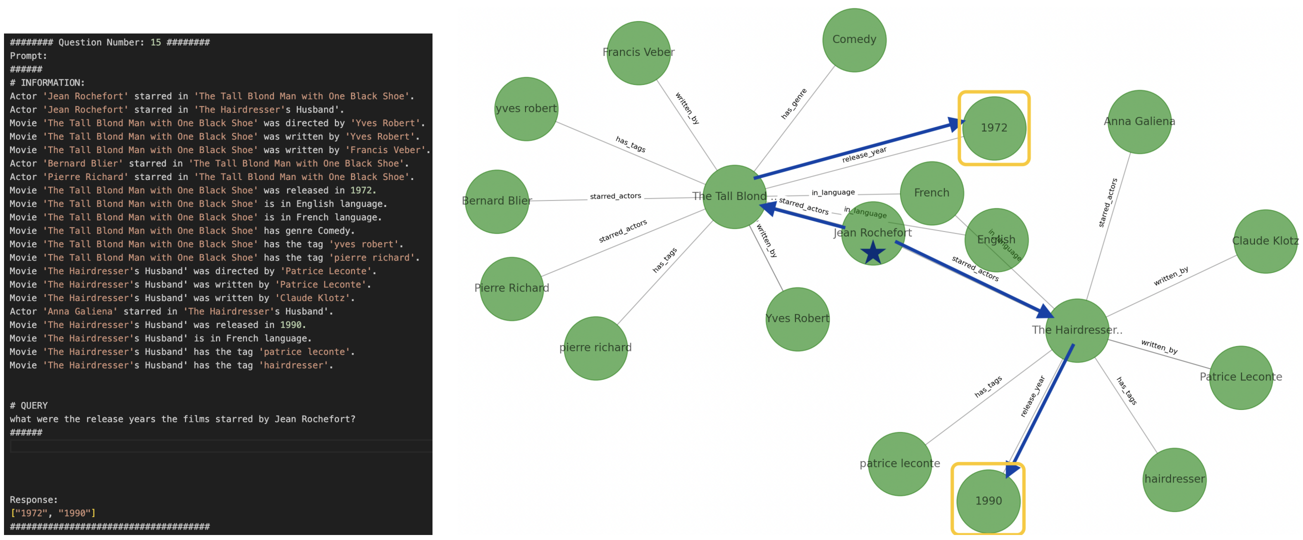
5. Results

To assess the language model's performance, a QA process was conducted using the MetaQA dataset. This evaluation included addressing a sample of 2,000 1-hop questions, as well as 1,000 2-hop questions and 400 3-hop questions. Each question in the dataset had one or more ground truth answers available. The language model was tasked with structuring its output into an array, where each element represented a potential answer. The model's objective was to populate this array with the most probable answers, leveraging its understanding of the question and the dataset information.

The reason for testing 2-hop and 3-hop setting with a smaller sample was that the size of information retrieved from the KG grew exponentially with the number of hops. Hence, due to the limited available tokens for our LLM input, the multi-hop experiments ran with a smaller sample set.

While basic accuracy metrics could serve for evaluation, a modified version of the F1-score was employed to assess the model's performance. This decision was made to accommodate scenarios where the model might generate answers not present in the ground truth or might fail to include all required correct answers. The F1-score provides a more balanced evaluation by considering both types of errors. Additionally, some questions posed challenges due to their intricate or incomplete sentence structures, regardless of the number of answers they might have. Such questions were challenging for the LLM to answer, since these models should be provided with a question in natural language and not just mentioning keywords. Hence, a fully correct answer to each question should contribute equally to the overall performance regardless of the number of answers required. This approach aligns with the inherent nature of language models and their performance in this task – either grasping the correct pattern and retrieving all answers accurately or failing to respond at all. Consequently, performance metrics were computed for each question individually and then aggregated (average) to determine the overall performance.

Specifically, to calculate the overall F1-score for each test set, both precision and recall rates were computed for each



```
######## Question Number: 15 ########
Prompt:
######
# INFORMATION:
Actor 'Jean Rochefort' starred in 'The Tall Blond Man with One Black Shoe'.
Actor 'Jean Rochefort' starred in 'The Hairdresser's Husband'.
Movie 'The Tall Blond Man with One Black Shoe' was directed by 'Yves Robert'.
Movie 'The Tall Blond Man with One Black Shoe' was written by 'Yves Robert'.
Movie 'The Tall Blond Man with One Black Shoe' was written by 'Francis Veber'.
Actor 'Bernard Blier' starred in 'The Tall Blond Man with One Black Shoe'.
Actor 'Pierre Richard' starred in 'The Tall Blond Man with One Black Shoe'.
Movie 'The Tall Blond Man with One Black Shoe' was released in 1972.
Movie 'The Tall Blond Man with One Black Shoe' is in English language.
Movie 'The Tall Blond Man with One Black Shoe' is in French language.
Movie 'The Tall Blond Man with One Black Shoe' has genre Comedy.
Movie 'The Tall Blond Man with One Black Shoe' has the tag 'yves robert'.
Movie 'The Tall Blond Man with One Black Shoe' has the tag 'pierre richard'.
Movie 'The Hairdresser's Husband' was directed by 'Patrice Leconte'.
Movie 'The Hairdresser's Husband' was written by 'Patrice Leconte'.
Movie 'The Hairdresser's Husband' was written by 'Claude Klotz'.
Actor 'Anna Galiena' starred in 'The Hairdresser's Husband'.
Movie 'The Hairdresser's Husband' was released in 1990.
Movie 'The Hairdresser's Husband' is in French language.
Movie 'The Hairdresser's Husband' has the tag 'patrice leconte'.
Movie 'The Hairdresser's Husband' has the tag 'hairdresser'.


# QUERY
what were the release years the films starred by Jean Rochefort?
######



Response:
["1972", "1990"]
####################################
```

Figure 2. A sample region of knowledge graph retrieved to answer Question#15 in the 2-hop questions.
The question asks the release year of the films that "Jean Rochefort" has starred in. It has been shown in the graph that the exploration starts at his corresponding node in the KG. It continues to finding the two movies that he has starred in, and then the release year can be retrieved through movies' nodes. On the right, it can be observed that all edge descriptions in this chunk of knowledge graph are provided to the LLM, whom is asked to retrieve these release years. Obviously, the task involved connecting disparate pieces of information and step-by-step reasoning.

Table 2. Model performance on each of the three settings. It is important to note that 7% of 3-hop questions caused the limited context window error.

Setting	Sample Size	Precision	Recall	**F1**
1-hop	2,000	0.9460	0.9706	**0.9581**
2-hop	1,000	0.8222	0.7351	**0.7762**
3-hop	400	0.5428	0.3965	**0.4583**

question separately. The average precision and recall rates across all questions were then calculated to derive the overall precision and recall scores. Subsequently, the overall F1-score was computed based on these scores.

Table 2 shows the model's performance in each task. As expected, the system showed superior performance in addressing 1-hop questions compared to 2-hop and 3-hop questions.

As demonstrated in Table 2, the model shows promising performance when addressing 1-hop questions. Moreover, it demonstrates relatively strong performance in the 2-hop setting compared to existing methods such as (Zhang et al., 2018) who proposed the dataset. However, the results for the 3-hop settings are less satisfactory. It is important to note that the system faced an error indicating exceeding LLM's context window when answering some 3-hop questions (specifically, 28 questions out of the sample of 400 3-hop questions $\rightarrow 7\%$).

This error means that the length of our input, including the question and a probably huge document of information retrieved from the KG, has exceeded the context window length limit of the utilized LLM, gpt-3.5-turbo. Although this LLM is amongst the best choices considering factors such as token price, context window, and model size, it still failed to fit the whole graph knowledge in its context window. The inclusion of nodes with high degrees in our knowledge graph, specifically in the neighborhood of some entities, caused such an issue.

6. Discussion

As anticipated, our model performed significantly better in the 1-hop setting. The major challenge for the LLM in this scenario lays in grasping the intent of the query and effectively retrieving the relevant information. For example, variations in how questions framed the concept of "tags" associated with movies posed challenges for the model's understanding in certain instances.

In the 2-hop setting, performance was reasonably acceptable despite some intricacies in the wording of questions. The decline in performance compared to the 1-hop setting was expected, given that the model must find connections between pieces of information dispersed across the provided documents and sentences. Refer to Figure 2, which illustrates the importance of step-by-step decision-making in multi-hop scenarios. Still, there remains significant room for improvement for the model in this setting. One potential approach involves prompting the LLM to demonstrate its reasoning process step by step, a strategy that has shown promise in similar research works. However, its implementation in our system poses challenges, particularly regarding parsing the output for performance evaluation.

Regarding the 3-hop setting, the limited context window error may justify the low success rate of our system. Although the input length for other 3-hop questions did not exceed the maximum limit of 16,385 tokens, it often approached this threshold. Despite approximately 93% of questions managing to contain their contextual knowledge within the context window, the length of input (LLM prompt) for many of them was notably long. This posed challenges for the LLM in synthesizing information scattered across the dataset and making informed decisions to answer the query.

7. Limitations & Future Work

One of the major limitations of this work was the limited context window of LLMs. As seen in the 3-hop tests, the information retrieved for some questions exceeded 16,385 tokens, the length of gpt-3.5-turbo context window. On the other hand, since the available tokens were limited, the edge description could not be as comprehensive as possible and had to be limited to a few words. Most importantly, this task dealt with LLMs, which can show highly unpredictable behavior in different scenarios. For instance, although the LLM was clearly instructed about the output format, many instances were observed in which the output format did not follow the guidelines and posed challenges in the parsing stage and evaluation process.

Conducting this work elaborated on many possibilities for future research. The LLM clearly struggled with step-by-step reasoning as it was more prone to mistakes when the questions required multi-hop reasoning. Consequently, building a framework that can guide the LLM in explaining its chain of thoughts and step-by-step answers may result in more accurate systems. Additionally, this work can prepare the building blocks for explaining graph-based retrieval-augmented generation (RAG) systems.

8. Conclusion

In this project, the integration of large language models with knowledge graphs was explored to enhance question-answering capabilities and address challenges associated with LLMs and recent techniques such as RAG. By providing LLMs with an external dataset for retrieval, augmented with the structured representation of knowledge offered by KGs, we aimed to improve the relevance of the provided information and, gradually, the relevance of the generated responses. Our system architecture focused on leveraging LLMs' linguistic proficiency to interpret user queries and retrieve relevant information from the knowledge graph. The system showed promising results, particularly in 1-hop scenarios. However, challenges arose in multi-hop scenarios, highlighting the need for further research into facilitating step-by-step reasoning and explanation generation by LLMs. Future research directions may include refining the system's ability to handle multi-hop reasoning, maximizing the utility of graph-based RAG systems, and developing frameworks for explaining LLM decision-making processes.

References

Abu-Rasheed, H., Weber, C., and Fathi, M. Knowledge graphs as context sources for llm-based explanations of learning recommendations. *arXiv preprint arXiv:2403.03008*, 2024.

Feng, C., Zhang, X., and Fei, Z. Knowledge solver: Teaching llms to search for domain knowledge from knowledge graphs. *arXiv preprint arXiv:2309.03118*, 2023.

Hagberg, A., Swart, P., and S Chult, D. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., and Wu, X. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108, 2019.

Saxena, A., Kochsiek, A., and Gemulla, R. Sequence-to-sequence knowledge graph completion and question answering. *arXiv preprint arXiv:2203.10321*, 2022.

Sun, K., Xu, Y. E., Zha, H., Liu, Y., and Dong, X. L. Head-to-tail: How knowledgeable are large language models (llm)? aka will llms replace knowledge graphs? *arXiv preprint arXiv:2308.10168*, 2023.

Wu, Y., Hu, N., Qi, G., Bi, S., Ren, J., Xie, A., and Song, W. Retrieve-rewrite-answer: A kg-to-text enhanced llms framework for knowledge graph question answering. *arXiv preprint arXiv:2309.11206*, 2023.

Zhang, Y., Dai, H., Kozareva, Z., Smola, A. J., and Song, L. Variational reasoning for question answering with knowledge graph. In *AAAI*, 2018.